VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

FACULTY OF MECHANICAL ENGINEERING

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

ŘÍDICÍ JEDNOTKA ČTYŘNOHÉHO ROBOTU NA BÁZI MIKROKONTROLÉRU PIC

CONTROL UNIT OF FOUR - LEGGED ROBOT BASED ON MICROCONTROLLER PIC

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — DANIEL YOUSSEF

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. ROBERT GREPL, Ph.D.

SUPERVISOR

BRNO 2013

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav mechaniky těles, mechatroniky a biomechaniky
Akademický rok: 2012/2013

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Daniel Youssef

který/která studuje v **bakalářském studijním programu**

obor: **Mechatronika (3906R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Řídicí jednotka čtyřnohého robotu na bázi mikrokontroléru PIC

v anglickém jazyce:

Control unit of four - legged robot based on microcontroller PIC

Stručná charakteristika problematiky úkolu:

Tato práce navazuje na úspěšný projekt kráčejícího robotu (ÚMTMB 2003, 2007). Hlavním cílem je vyrobit novou řídicí jednotku na bázi mikrokontrolerů PIC, která bude využívat pokročilou sensoriku včetně sensoru síly v kontaktu nohy s terénem.

Cíle bakalářské práce:

1) Doplnění stávající konstrukce robotu o snímače síly v kontaktu s terénem.
2) Návrh a výroba řídicí jednotky na bázi PIC32. Součástí jsou sensory (akcelerometr, gyroskop), dotykový displej a komunikační linky pro další rozšiřování funkcionality (SPI, CAN, RS232).
3) Implementace dodaného kinematického modelu pro PIC32, testování.
4) Implementace jednoduchého algoritmu chůze robotu, který bude používat kinematický model (chůze dopředu po rovině, zatáčení, chůze po mírně nerovném terénu).

Seznam odborné literatury:

- Valášek, M.: Mechatronika, Vydavatelství ČVUT 1995
- Grepl, R.: Modelování mechatronických systémů v Matlab/SimMechanics, BEN, 2007
- Mann, B.: C pro mikrokontroléry, Nakladatelství BEN, 2003
- Herout, P.: Učebnice jazyka C
- Janíček, P., Ondráček, E.: Řešení problémů modelováním, skriptum VUT Brno, 1998
- Horáček, P.: Systémy a modely, ČVUT 1999
- Grepl, R.: Využití komplexních dynamických modelů při návrhu a řízení kráčejícího robotu, disertační práce, ÚMTMB FSI VUT v Brně, 2004
- Bezdíček, M., Grepl, R., Švehlák, M., Chmelíček, J.: Artificial Neural Network Application To Walk Of A Four Legged Robot, Inženýrská mechanika 2004
- Švehlák, M., Grepl, R., Věchet, S., Bezdíček, M., Chmelíček, J.: Design Of Small Laboratory Quadruped Robot, Inženýrská mechanika 2004

Vedoucí bakalářské práce: doc. Ing. Robert Grepl, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2012/2013.

V Brně, dne 19.11.2012

L.S.

prof. Ing. Jindřich Petruška, CSc. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c. Děkan fakulty

Abstrakt:

Tato práce doplňuje konstrukci "Jaromíra", čtyřnohého robota,vhodnými senzory a přidává novou řídící jednotku,která bude schopná přijímat data ze senzorů a vhodně je zpracovávat. Správná funkce řídící jednotky bude otestována při řízení několika jednoduchých pohybů.Tento test demonstruje možnost použít tuto jednotku pro vývoj a testování komplexních algoritmů pohybu robota.

Klíčová slova:

Robot, I2c, servo motor, senzor, mikrokontrolér.

Abstract:

This Bachelor thesis develops “Jaromir“, a four legged robot by adding appropriate sensors to it and designing a new control unit that is able to receive and process the acquired data by the sensors. The functionality of the control unit is tested by implementing a number of simple movements. This testing demonstrates the possibility of using this unit for the developing and testing of complex movement algorithms.

Key words:

Robot, I2c, servo motor, sensor, microcontroller.

Bibliografická citace:

YOUSSEF, D. Řídicí jednotka čtyřnohého robotu na bázi mikrokontroléru PIC. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 40s. Vedoucí bakalářské práce doc. Ing. Robert Grepl, Ph.D..

Čestné prohlášení:

Prohlašuji, že jsem bakalářskou práci vypracoval Samostatně s použitím odborné literatury a pramenů uvedených v seznamu, který tvoří přílohu této práce.

Květen 2013 ……..

Daniel Youssef

Poděkování:

Děkuji doc. Robertovi Greplovi, ing. Vojtěchovy Lamberskému a ing. Josefovi Vejlupkovi Za jejich rady a pomoc. Taky děkuji všem kolegům za příjemné pracovní prostředí v Mechlabu.

Contents

1 Introduction:

For any Robot, acquiring information about the surrounding environment is essential, and can be really helpful when it comes to designing movement algorithms. This thesis' goal is to be a foundation stone for any upcoming job that will aim to build movement algorithms depending on captured data about the robot state. Various sensors can be implemented including contact sensors, force sensors and current sensors, which imply choosing the appropriate sensors as a start. The next step will be designing and manufacturing a control unit whose brain is going to be microcontroller PIC that will be able to receive data from the sensors, and process them. In order for the microcontroller to properly interact with the received data the inverse kinematics model of the robot should be implemented. The final stage will be designing some basic algorithms that will test the functionality of the control unit. The goal of these algorithms is to enable the robot to accomplish some simple actions (standing up, sitting down, turning around, walking.....). The following text is divided into two main sections:

1- Used instruments and their properties: an introduction to the problems this thesis is trying to solve in addition to research for the devices and tools used to deal with these problems.
2- Solution procedures: choosing and implementing the appropriate tools.

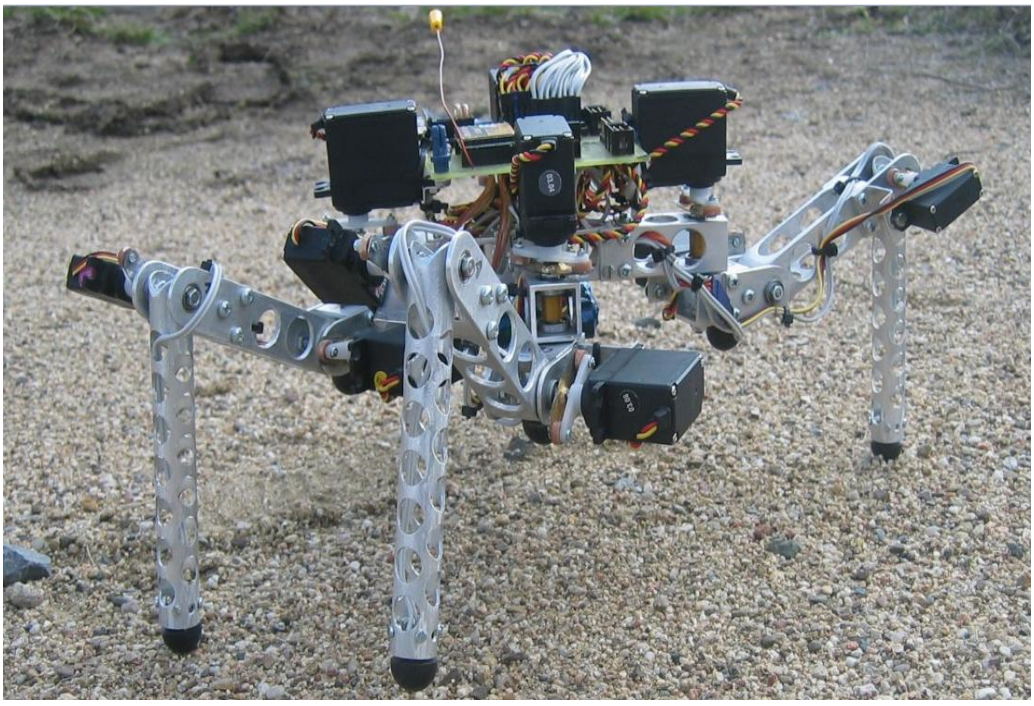


Figure (1) Jaromír

2 Used instruments and their properties:

2.1 Force sensors:

The first problem that had been encountered was how to obtain data referring to the state the robot is facing. The basic choice was to use force sensors that can measure the force each leg is undergoing, and transform it to a signal that will be fed to the control unit, which will give a good estimation of the robot state. Force sensors are too many and the following chapters will introduce you to some of the sensors considered to be used.

2.1.1 Piezoresistive sensors:

These sensors measure pressure or strain by different techniques but all are based on the principle of resistance variation according to the applied load.

2.1.1.1 Strain gauge:

A device used to measure the strain or pressure applied to a particular body through simple change in resistance. Strain gauges are usually built from a thin wire shaped in a way that would increase the part undergoing the applied pressure. When strain or pressure is applied, the wire is extended or compressed which means a change in resistance. Mostly, strain gauges are used in Wheatstone bridge configuration so the change in resistance is reflected as a change in voltage linear to the pressure or strain applied [6].

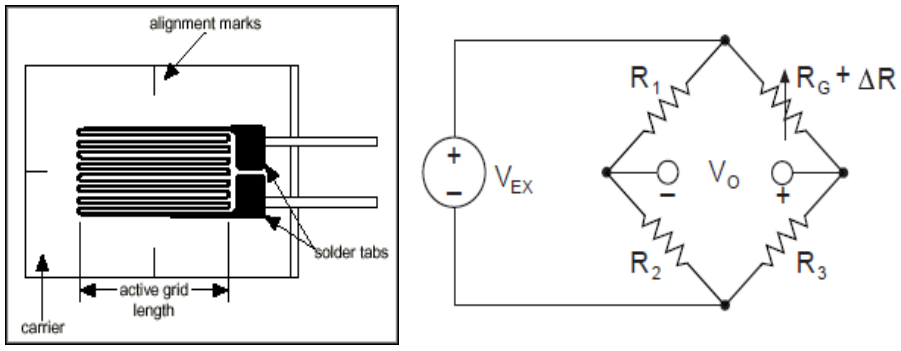


Figure (2) an illustration of strain gauge and attaching it to a Wheatstone bridge [6]

2.1.1.2 The Fss Sensor:

This sensor measures force using a piezoresistive silicon element that changes its resistance according to the applied pressure. The metallic ball on top of the sensor helps to concentrate the force applied and delivers it to the sensing silicon element .A Wheatstone bridge has been implemented into the sensor, which stabilizes the output voltage.

It is used mainly for pressure sensing and the applied weight can vary from 0 to 1500 grams with a sensitivity of 12mV/g [7].

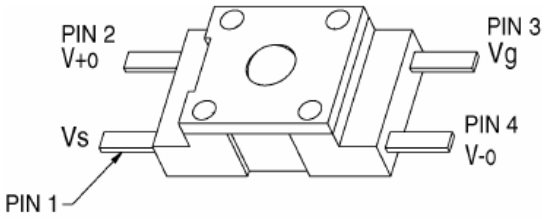


Figure (3) Fss sensor pin mapping [7]

2.1.2 Flex sensors:

This sensor is made out of a resistive carbon element, which can change resistance according to the bending, the bigger the bending angle the higher the resistance. This sensor can be implemented into a number of electrical circuits in order to get from the changing resistor a change in voltage, which can be used as a helpful signal in many applications such as robotics [8].

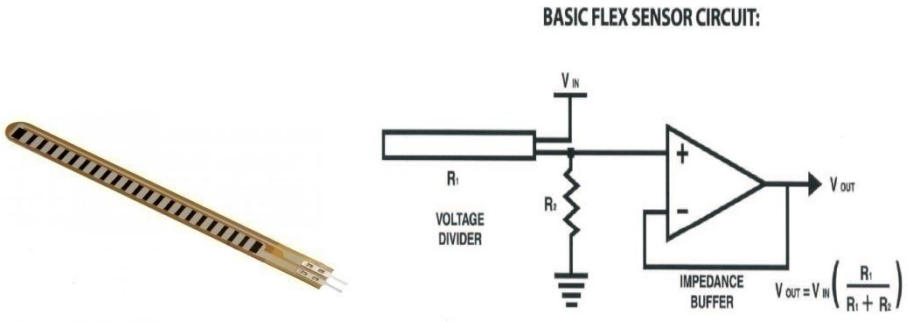


Figure (4) illustration of flex sensor and basic flex sensor circuit [8]

2.1.3 Piezoelectric sensors:

These sensors are built from special materials such as quartz and many ceramic and plastic materials that have the ability to generate different amount of electrical charge according to the amount of the applied pressure. Due to the fact that their output signal may noticeably decrease with time, these sensors fit much better for dynamic applications than for static ones. This is caused by the internal impedance of the sensor and the impedance of the output circuit.

2.1.3.1 Quartz force rings (3 component force sensor):

These sensors have a greater advantage over common force sensors, which is the ability to measure all three force components with a wide measuring range, and work in really rough circumstances, thanks to their strong casing. Each sensor consists of three pairs of quartz rings, two quartz pairs are able to measure shear force (Fx,Fy components) and the third pair measures pressure (Fz component). Three electrical charges are generated, each proportional to the corresponding force component [9].

techniques	Technical Data*		
Range	F_x, F_y	kN	–2,5 ... 2,5
	F_z	kN	–5 ... 5
	F_z	kN	0 ... 30
Overload	F_x, F_y	kN	–3/3
	F_z	kN	–6/6
	F_z	kN	36
Threshold		N	<0,01
Sensitivity	F_x, F_y	pC/N	≈ –8
	F_z	pC/N	≈ –4

Figure (5) Kistler 9251A and some Basic parameters [9]

2.2 Current sensors:

Using Force sensors hit a dead end (reasons will be explained later), so searching began for another way of capturing data, and current sensors seemed as a good alternative. These sensors measure current intensity that each servo motor consumes in any given moment, which will determine the applied moment on each motor, and give the previous estimation of the robot state. Industry has gone ahead a long way in manufacturing current sensors, which are now available in relatively small packages and have really good sensitivity. The following will acquaint you with some of these sensors:

2.2.1 Hall Effect sensors:

The Hall Effect principle depends on the fact that a difference in voltage is produced due to the mutual influence of current and a perpendicular magnetic field. This principle is used as an efficient method to measure the current passing through a conductor or the magnetic field generated due to this passing. Sensors that use this principle are called Hall Effect sensors. One of them is:

2.2.1.1 ACS712:

This AC/DC current sensor is manufactured by Allegro, a well known firm when it comes to current sensing. This sensor mainly consists of A Hall circuit and a copper conductor. The measured current passes through the conductor, thus generates a magnetic field proportional to the current intensity. This magnetic field is measured by the Hall circuit, which produces a corresponding output voltage that can be used as an indicator to the intensity of the measured current [10].

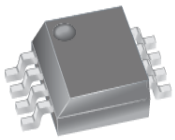


Figure (6) the ACS712 sensor [10]

2.2.2 Current sensing resistor:

The following method is one of the most basic techniques used for current sensing. A resistor is placed in series with current path, and information about current intensity is obtained through measuring the voltage drop across the resistor using Ohm's law. Although this method of current sensing seems very simple, a number of precautions must be taken into account, such as the value of the resistor which must be as low as possible, so the voltage drop will not be that big which can affect the whole circuit. Also we should consider the maximum power that the resistance consumes, in order not to exceed the maximum value permitted, which can be done by a good estimation of the maximum current passing through the resistor. Time constant:

$$\tau_a = L / R \qquad (1)$$

L : is the parasite induction of the resistor.
R : is the value of the resistor.

if fairly large, can cause many problems including distortion of the output voltage, so we should choose the resistance to have little inductance (L), metallic resistors for example [5].

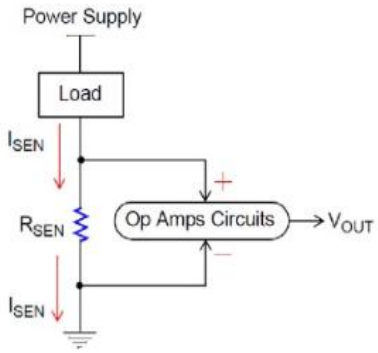


Figure (7) illustration of current sensing resistor technique

2.3 Control unit:

After obtaining the needed data, the next step is processing them by the control unit, which should contain a microcontroller able to receive data from the sensors, process them according to the downloaded program and then operate the robot servo motors according to the results. The following will acquaint you with the main components of this control unit:

2.3.1 Microcontroller:

It represents the brains of the control unit. Sensors´ Data are fed to the microcontroller through the analog inputs it has, which have usually a voltage range (0....3.3 V) .The analog input signals are transformed to digital form, required by the microcontroller, through ADC convertors. Microcontrollers have many features including I2c interface, SPI interface, and UART communication and CAN communication that can be used to communicate with various external devices (sensors, servo drivers ……..).

2.3.2 Connecting sensors to microcontroller:

The next step in designing the control unit is finding a way to get the outputs of both force and current sensors to an acceptable level for the analog inputs of the Microcontroller. Because of the low output voltage that we get in both cases an operational amplifier is used .The following will acquaint you with the connections of OP used in the control unit.

2.3.2.1 Non-inverting amplifier:

This connecting of the operational amplifier allows to amplify the input voltage without inverting it (The output voltage will still have the same polarity) [4].

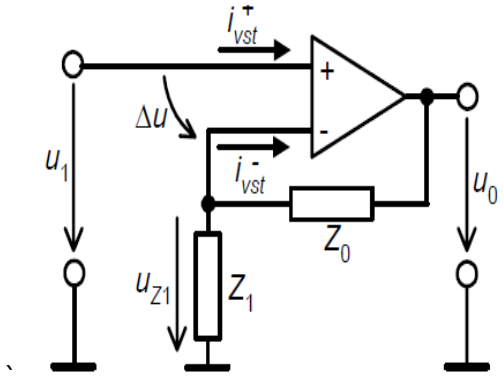


Figure (8) non-inverting operational amplifier [4]

$$U_{out} = (\frac{Z_0}{Z_1} + 1).U_1 \qquad (2)$$

U_0 : is the output voltage.

U_1 : is the input voltage.

Z_0 , Z_1 : are usually resitors.

2.3.2.2 Differential amplifier:

A special connection of the operational amplifier is used to amplify and convert a differential voltage to a normal one. The input of the O.P is a differential voltage that may be created by two supply sources, or simply by measuring voltage drop between two points of a circuit none of them is ground, and its output is measured between output point and the ground of the circuit [4].

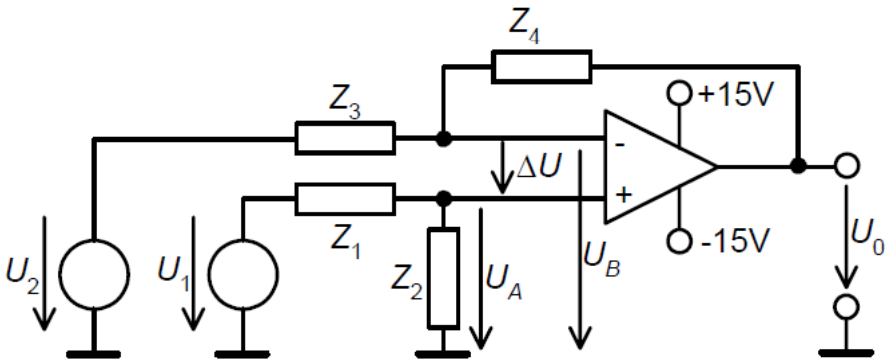


Figure (9) Differential amplifier [4]

In order for the last circuit to work as a differential amplifier the following conditions should be true:

$$z_1 = z_3 \qquad z_2 = z_4 \qquad (3)$$

Then the output voltage will be:

$$U_{out} = \frac{Z_4}{Z_3}(U_1 - U_2) \quad (4)$$

U_{out} : is the output voltage.

$U_1 - U_2$: is the differential input voltage.

Z_1, Z_2 , Z_3 , Z_4 : usually resitors.

2.3.3 Batteries voltage measuring:

An important function that the control unit should be able to perform is giving information about the state of the batteries feeding it (U1). This could be done in a number of ways including a simple voltage divider. Its output (U2) is connected to one of the analog inputs of the microcontroller. The values of the resistors should be carefully calculated in order that the microcontroller analog input will not exceed 3.3 V at maximum voltage of the batteries.

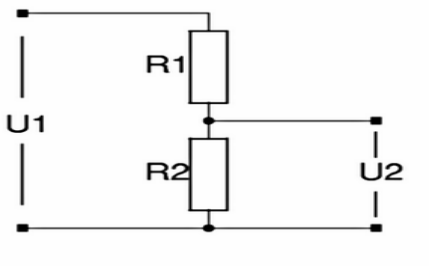


Figure (10) voltage divider circuit

2.3.4 Multiplexer:

Although advanced microcontrollers have a big number of analog inputs, receiving data from the 12 current sensors, 4 Force sensors and the voltage divider would require all if not more inputs than available. One way to make this connection possible is using analog multiplexing for the 12 current sensors. The following will acquaint you with some Analog multiplexers:

2.3.4.1 AD8174:

Besides the ability of this analog multiplexer to choose one of its four analog inputs and deliver it to its output, it contains an operational amplifier right before its output that is in the non-inverting mode, whose amplification can be set by external resistors. This feature can be helpful in many cases [11].

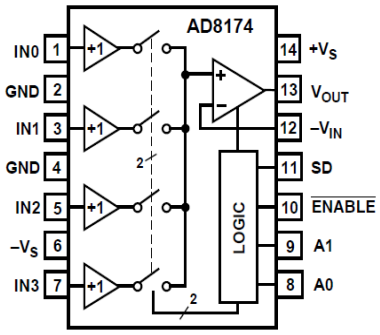
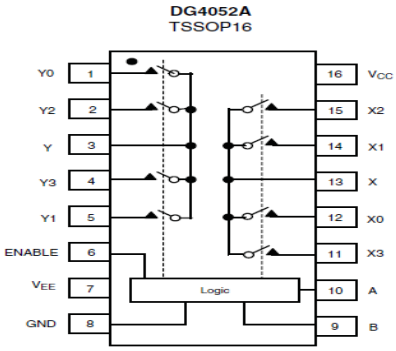


Figure (11) AD8714 internal blocks [11]

2.3.4.2 DG4052A:

This device is dual 4 input analog multiplexer. It has eight inputs and 2 outputs and only two digital inputs for output selection [12].



Enable	A	B	Output
H	X	X	All switches are open
L	L	L	Y to Y0,X to X0
L	H	L	Y to Y1,X to X1
L	L	H	Y to Y2, X to X2
L	H	H	Y to Y3,X to X3

Figure (12) Dg4052A internal blocks and truth table [12]

2.3.5 Servo driver:

The connection issue between all the sensors and the microcontroller has been solved. The next step will be to design the second part of the control unit, which is responsible for transporting data from the outputs of the microcontroller to the servo motors. The robot has 12 servo motors, which implies that the microcontroller should be able to generate 12 independent PWM signals, which is too many even for advanced generations. That is why a servo driver is needed, which mostly is a preprogrammed chip that uses I2C or SPI interface to communicate with microcontroller.

2.4 Programming:

The final stage of this bachelor thesis is programming the microcontroller in order to enable the robot to accomplish some basic tasks. Most of the programming was done in Matlab Simulink environment using a special embedded toolbox to provide the required interface between the Simulink environment and the microcontroller. The next chapter will acquaint you with the I2c protocol that was used to communicate between the microcontroller and the servo driver.

2.4.1 I2c communication (internal integrated circuit):

It was invented in order to enable motherboards and microcontrollers to communicate with low speed devices. It uses only two bidirectional lines SDA(serial Data line), and SCL(serial clock lines) connected to pull up resistors, and supplied by a voltage that usually ranges between 3.3V and 5V .This interface has the ability to link one master with a number of slaves .Each salve has a special address, that consists of 7 or 10 bits. Each exchange of data begins with a start bit, which is accomplished by changing the state of SDA line from high to low, while SCL is set high. The next step is sending the first byte, which contains the 7 bit address of the slave and one last bit to determine whether the operation is reading or writing (write to slave [0], read from slave [1]). If the slave is connected it sends an acknowledgment bit, which starts data exchange. If the master is writing, it sends one byte to the slave and waits for an acknowledgment bit, receiving this bit initializes a new sending cycle. If the master is reading, it gets one byte every time it sends an acknowledgment bit to the slave. Notice that all the changes in the SDA line after the start bit are done with SCL set low. The data exchange is done after a stop bit, which is accomplished by changing the state of SDA line from low to high while setting the SCL line high [13].

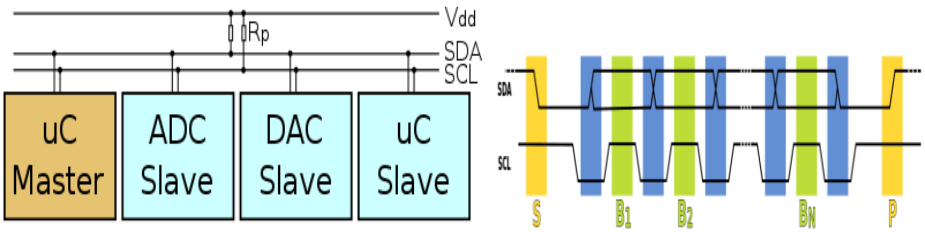


Figure (13) I2C interface and its time diagram [13]

3 Solution procedures:

3.1 Force sensors applications:

In the previous section we have been acquainted with a number of force sensors. Their job was to obtain information about the force exerted at each leg. The next chapters will explain the reasons lying behind choosing or excluding each sensor.

3.1.1 Strain gauge and flex sensor possible implementation:

The main idea was to use strain gauge as pressure sensor at each leg. The gauge would be implemented in Wheatstone bridge, and the output voltage would be proportional to the applied pressure. The robot can face walking on a non-flat terrain, where pressure calculated by the strain gauge would not be sufficient to give an idea about the robot state. Here comes the role of flex resistor, which will be attached to the thigh joint that rotates in a vertical plane. The output voltage of the circuit attached to the flex sensor varies according to the bending of the joint, when the radius of the bending is bigger the output voltage is smaller. Both voltages are fed to the microcontroller analog inputs. Because of the fact that both sensors need to be attached to an electrical circuit, in order to get the required change in voltage, and due to the imprecision this technique suffers, this method was excluded, and search began for some integrated sensors that will be able to do the job.

3.1.2 Quartz force rings compatibility:

As mentioned before this sensor has the ability of measuring the three components of exerted force, with high precision and wide range, which would be an ideal solution if it was not for its high price that was not suitable for this project.

3.1.3 Fss sensor testing and mounting:

This sensor's reliability, suitable measuring range, small packaging, the ability to concentrate pressure through its metallic ball and acceptable price were the determining facts behind choosing it. The first step was putting the sensor under experiment, and confirming the information published by the manufacturing firm. The real time card MF624 was used as a link between the sensor and Matlab/Simulink environment. A simple model was programmed to provide a supply voltage for the sensor and receive the output voltage from it. Different pressures were applied to the sensor while the model was storing data about the output voltage.

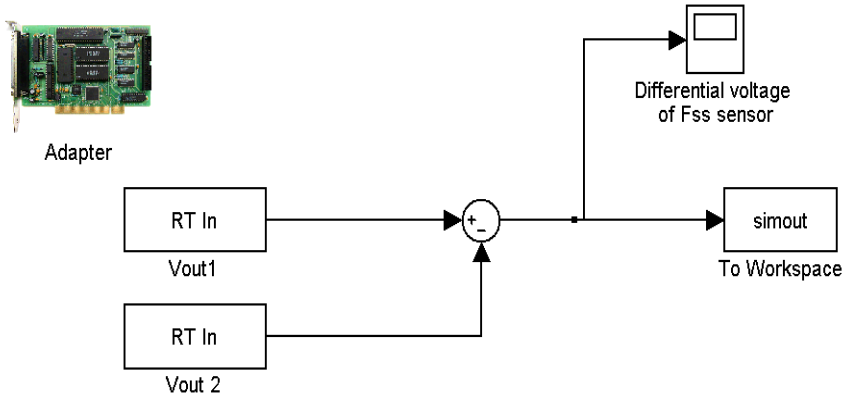


Figure (14) Simulink model for extracting data out of the Fss

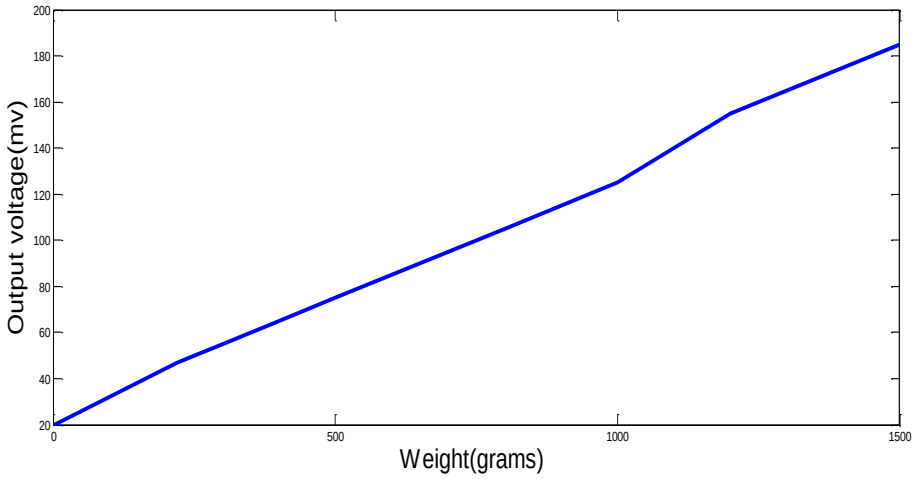


Figure (15) weight/output voltage characteristic of the Fss sensor

The previous characteristic demonstrates the fact that the output voltage of the Fss sensor is linearly proportional to the applied pressure. After measuring, we faced the problem of attaching the sensor to the robot leg, which seemed to be fairly complicated due to the design of the robot and the fact that this sensor did not give the right results, when the pressure was not perpendicular to the metallic ball, which was a case to occur often during the process of walking even on a flat terrain. At that time we started thinking about more dependable way to measure Forces exerted at the legs without excluding the use of the Fss sensors.

3.2 Current sensors as a sensing alternative:

In the research section two basic current sensing techniques were explained. The following will provide you with the primary reasons that were behind choosing or excluding each technique.

3.2.1 Reasons prevented from using Hall Effect sensors:

Although these sensors are really dependable, come in small packages, and they are not expensive, they were not used. All integrated circuits that use the Hall Effect principle such as the ACS712, which I came across, were designed to measure one current that would mean using 12 integrated circuits in order to measure the currents of all 12 servo motors, which would be considered rather expensive, and would really make the control unit much bigger.

3.2.2 Current sensing resistor parameters:

After excluding the Hall Effect sensors, a simpler way of measuring the current of the servo motors was considered. A current sensing resistor seemed as a good solution that was not expensive, and would not need much space on the control unit. In order to choose the right resistor for the measuring, the maximum current consumed by the servos was measured. It was about (0.6 A) (in the calculations 0.75 A was used as a maximum value). SMD resistors were considered, so the parasite induction of the resistor would have a miner effect, and the time constant τ_a would be as small as possible. The smallest available value of SMD resistor with maximum power consumption of (0.25 W) was ($0.33\,\Omega$), which means:

$V_{\max} = I.R = 0,75.0,33 = 0,247v$ (Small voltage drop) (5)

$V_{\min} = I.R = 0,1.0,33 = 0,033v = 33mv$ (Acceptable as OP input) (6)

$P_{\max} = I^2.R = 0,75^2.0,33 = 0,185w$ (Acceptable) (7)

$V_{\max}$: is the maximum voltage drop across the resistor.

$V_{\min}$: is the minimum voltage drop across the resistor.

$P_{\max}$:is the maximum power consumed by the resistor.

I : is the current passing through the resistor.

3.3 Control Unit:

3.3.1 PIC32MX:

PIC32MX795F512L is an advanced 32 bit microcontroller that supports many ways of interfacing. It has a big number of multiplexed analog inputs, and it is supported by Kerhuel Embedded tool box. All the features above played a big role in choosing this microcontroller as the brains of the control unit.

Note: The designed control unit is divided into two boards. First board contains the PIC32 microcontroller, and the second contains the rest of the control unit. Due to incorrect generation of C code for the PIC 32 microcontroller by the used embedded toolbox, it was replaced by 16-bit microcontroller dspic33Fj128Mc802, for which the mentioned toolbox has a better support.

3.3.2 Connecting sensors to PIC through OPs:

As mentioned before the output voltage of both current sensors and force sensors is too small to be fed directly to the microcontroller, so we need to use operational amplifiers to get it to a level that suits the range of the A/D converter.

3.3.2.1 Current sensors connection through non-inverting amplifiers:

The output voltage of the current sensors ranges from (0V) to (0.264V) and is measured across each resistor. The output voltage will be fed through a suitable RC filter with a medium time constant ($\tau = R.C$) to the input of an O.P in the non-inverting mode.

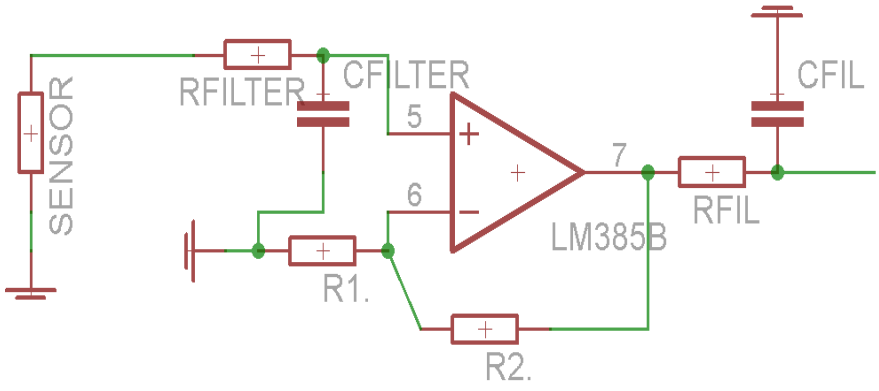


Figure (16) amplification of current sensor's output voltage

The output voltage of the O.P is also filtrated, and fed to the microcontroller. The output should have a maximum value of 3.3 V (the max. value of the PIC input).By making an easy estimation, we assume that the O.P amplification which is used in the non-inverting mode will be (K=R1/R2=12), and we choose R1 to be (1K), thus R2 is

going to be (12 k). Here is a simple calculation of the maximum and minimum output voltages:

$$V_{out\max} = (1 + R2/R1).V_{in\max} = 13.0,247 = 3,2V \text{ (Acceptable)} \quad (8)$$

$$V_{out\min} = (1 + R2/R1).V_{in\min} = 13.0,033 = 0,43V \text{ (Acceptable)} \quad (9)$$

$V_{out\max}, V_{out\min}, V_{in\max}, V_{in\min}$: maximum and minimum input, output voltages of the OP. $R1, R2$: resistors connected to the OP.

The filter should have a resistor of a small value up to (1K) and a capacitor of a big value around (100nF), so that noise effect would be minimal [5].

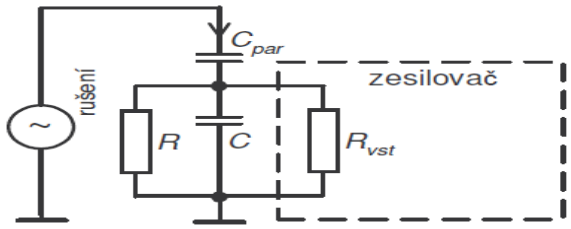


Figure (17) illustration of noise effect on filtration [5]

3.3.2.2 Force sensors connection through differential amplifiers:

The output voltage of the Force sensor ranges from (20 mV) to (125 mV). Same filtration used in the previous chapter is connected to the input, and the output of the operational amplifier. Because the output voltage of the Fss sensor is differential, we cannot use the OP in the non-inverting mode, but we use it as a differential amplifier.

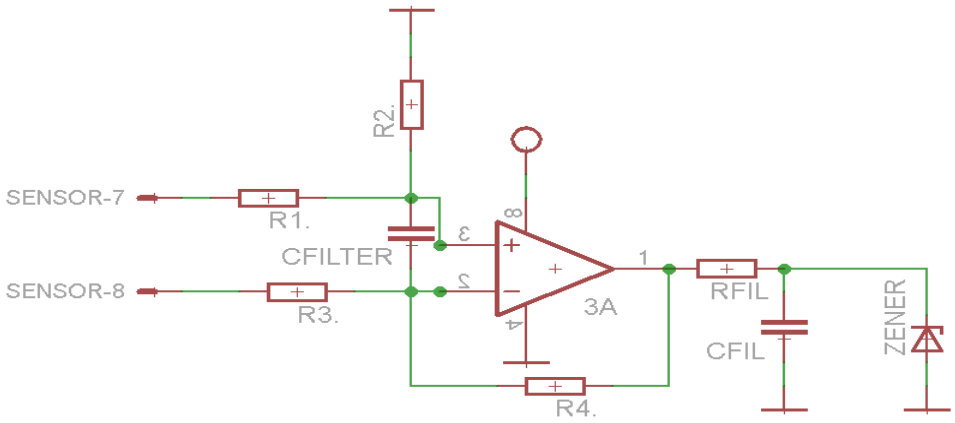


Figure (18) amplification of force sensors output voltage

A simple estimation suggests that the O.P should have an amplification about (K=R4/R3=R2/R1=24). We choose the values of the resistors to be (R3=R1=1K) so (R2=R4=24k).

$$V_{out\max} = (R2/R1).V_{in\max} = 24.0,125 = 3V \text{ (Acceptable)} \quad (10)$$

$$V_{out\min} = (R2/R1).V_{inmin} = 24.0,02 = 0,48V \text{ (Acceptable)} \quad (11)$$

Notice that after the RC filter at the output, a Zener diode is connected to protect the microcontroller from any voltage rise that could be caused by a failure of the operational amplifier. The operational amplifier Used in both cases is LM358. It is a rail to rail OP (needs only one supply voltage), and has a small offset.

3.3.3 Voltage divider as a simple batteries state indicator:

The research section mentions using a voltage divider to measure the state of the batteries. The output of the divider is fed to an analog input of the microcontroller .Therefore we should make sure that the maximum output voltage that corresponds to the maximum voltage of the batteries does not exceed 3.3V.

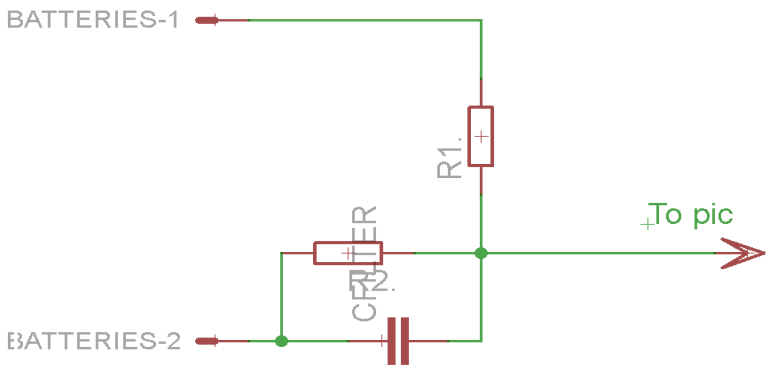


Figure (19) Batteries voltage measuring

We are using 6 rechargeable batteries each has a maximum voltage of (1.4 V). By a simple estimation we choose (R1=10k) and (R2=5.6 k).

$$V_{bat\max} = 1,4.6 = 8,4V \quad (12)$$

$$V_{out\max} = V_{bat\max} \cdot \frac{R2}{R1+R2} = 8,4\frac{5,6}{10+5.6} = 3,02V \text{ (Acceptable)} \quad (13)$$

$V_{bat\max}$, $V_{out\max}$: maximum voltage of the batteries and the ADC input.

$R1$, $R2$: resistors that form the voltage divider.

3.3.4 Applications of the analog multiplexer:

The 16 analog inputs of the microcontroller cannot cover all the input signals needed to be processed. This is what made us use analog multiplexing with the current sensors signals. Two different kinds of multiplexers were considered and the following explains why one of them was preferred over the other.

3.3.4.1 AD8714:

Although this multiplexer contains an O.P, which is needed for the amplification of the output voltage of the sensors, its high price and inaccessibility prevented from choosing it.

3.3.4.2 DG4052A:

Its reasonable price, small packaging, accessibility, ability to receive really small inputs, and being a dual 4*1 multiplexer lie behind selecting this multiplexer to be a part of the control unit. The outputs of the current sensors that measure currents of the motors on each leg are fed to one multiplexer. The outputs of all four multiplexers are controlled by the microcontroller through two digital outputs that are connected to the selection lines of the multiplexers, which means that at a given moment the microcontroller will read data from only four motors that have the same position on each leg. Small intervals specified by the state changing of selection lines will determine when the microcontroller will move to read other data.

3.3.5 The SD20 Servo driver:

The SD20 seemed as a good solution to the small number of PWM outputs that PIC 32 provides. It is preprogrammed PIC16F872 chip that has the ability to drive 20 servo motors and it communicates with the microcontroller through I2C interface [14].

Figure (20) The SD20 servo driver [14]

3.4 Implementation of the kinematic model into the microcontroller:

Programming any movements of the robot requires using the inverse kinematics block for an instance, so what is inverse kinematics? It is the problem of calculating the joints' angles for certain coordinates of the end effecter in the Global coordinate system [1]. The inverse kinematics block receives the X,Y,Z coordinates of the end of each leg, and calculates the corresponding rotation angles of each servo motor to reach the required position.

The previous kinematic model was developed in Matlab 2012a, whereas the intended programming was in 2009th version, which contains special embedded toolbox that enables the programming of various microcontrollers including pic 32 directly from Simulink environment. The first step was to modify the kinematic model to fit the 2009^{th} version, which was done through replacing some structures by normal variables, which led to losing some loops in the program. The second step was the implementation of the model into the microcontroller and testing the ability of this advanced microcontroller to deal with the big amount of float number calculations this model uses, which was done through using the overload flag in Kerhuel toolbox, which is set to logical one, if the execution of a given time step is not done when the following time step starts. Unfortunately the overload flag was on, which ensured that the microcontroller is slow to deal with the floating point calculations implemented in the model, although a big step size that reached 0.1 second was used in the Simulink model.

In order to solve this problem the main model was divided into two sub models:

1- A model that includes the inverse kinematics and calculates the required rotation angles.
2- A model for communication with the servo driver that uses the results from the first model.

Only the second model was intended to be implemented into the microcontroller, which clearly reduces the amount of calculations the microcontroller should deal with. To accomplish the previous task two Simulink blocks were used: 1- to work space 2- from workspace.

Besides using these two blocks to transfer data between both models, the second block played a crucial role in interpolating the acquired data to get a smooth movement of the robot. Because of the limited data memory the microcontroller has, a small step size could not be used in the first model, which would have generated a big amount of data that the microcontroller would not be able to allocate memory for. That's why a big step size was used, after that the data were interpolated in the

second model through the From Workspace block, using a linear interpolation, which the block provides.

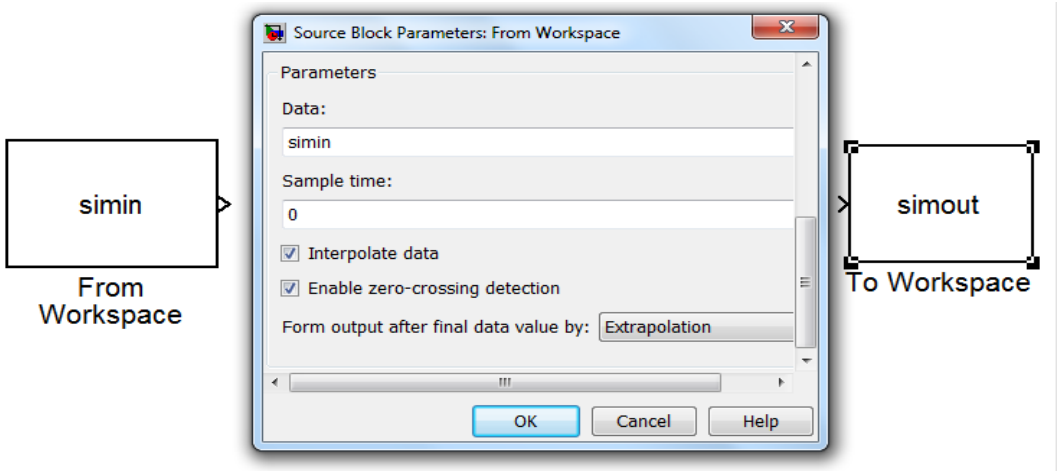


Figure (21) To work space block, From work space block and its parameters

3.5 Programming simple movements of the robot:

3.5.1 I2c communication between the servo driver and the microcontroller:

The communication between the servo driver that drives all the 12 servos of the robot and the microcontroller as mentioned before is done through I2c protocol. Three blocks that represent functions wrote in C Language and implemented into Matlab designed by Matěj Šimurda, were used for the I2c communication.

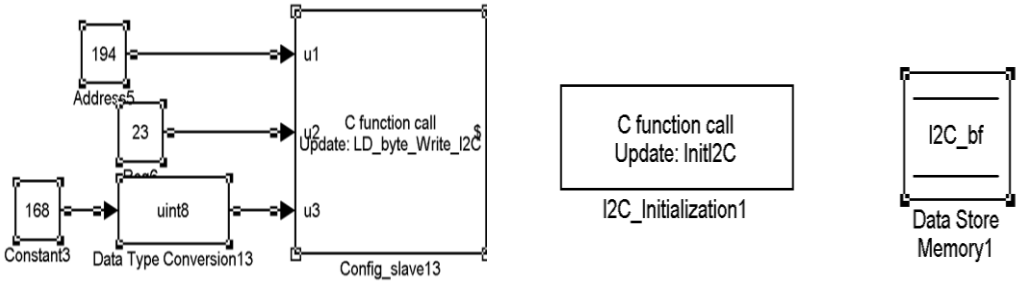


Figure (22) I2C writing, initialization and Data store Memory blocks

The Data writing block has three inputs: 1- the address of the controlled chip on the I2c lines (which is 194 for SD20 servo driver).2- the number of the register, we want to write on 3- the number we want to send to the register.

The SD20 servo driver has 23 registers: register 0 contains the software revision number. Registers 1 to 20 are used to drive the possible 20 servo motors connected to the driver. Register 21 is for the standard /expanded mode. Registers 22-23 are the offset of the expanded mode.

3.5.1.1 Testing the I2c communication in the standard mode:

We download a simple testing program to the microcontroller with number 0 written to register 21 (standard mode). Pulse generator is connected to registers 1, 10 to 20 that drive the servos of the robot. The pulse generator changes between 1 and 255 to test the range of each servo motor. The range of both thigh servo motors is 180 degrees. Standard mode covers only half of the possible range which means that we should use expanded mode to get the desired range. The following table shows the experimental results of the standard mode:

Angle	Status
-90 to -50	Out of range
-50 to 50	Within range
50 to 90	Out of range

3.5.1.2 Testing I2c communication with expanded mode:

Previous experimental results show us that pulse range should almost be doubled. Standard mode (1ms to 2ms), where as desired range might be (0.5ms to 2.5ms) as an estimation. The numbers we need to write to registers 21, 22, 23 are calculated as following:

$$reg22:23 = 500 - 20 = 480\mu s \quad (14)$$

$$(480)_{decimal} = (1,1110,0000)_{binary} \quad (15)$$

$$(1110,0000)_{binary} = (224)_{decimal} \quad (16)$$

$$reg21 = \frac{255*256}{rw} = \frac{255*256}{2000} \approx 33. \quad (17)$$

$$1 \rightarrow register22, 224 \rightarrow register23, 33 \rightarrow register21 \quad (18)$$

$reg21$, $reg22$: registers 21 and 22.
rw: range width.

The previous values are written to registers 21, 22, 23 in order to get the required pulse range. We did the previous testing again after writing the calculated values to the corresponding registers. Unfortunately the motors hit their stops, which implies that pulse range should be reduced. After a number of tries we got suitable pulse range, which is 1.6(0.7 to 2.3). The numbers that should be written to the registers 21, 22, 23 are calculated in the same way:

$$reg22:23 = 700 - 20 = 680\mu s \tag{19}$$

$$(680)_{decimal} = (10,1010,1000)_{binary} \tag{20}$$

$$(1010,1000)_{binary} = (168)_{decimal} \tag{21}$$

$$reg21 = \frac{255*256}{rw} = \frac{255*256}{1600} \approx 41. \tag{22}$$

$$2 \rightarrow register22, 168 \rightarrow register23, 41 \rightarrow register21 \tag{23}$$

$reg21, reg22$: registers 21 and 22.
rw : range width.

3.5.1.3 Connecting kinematic model to the registers:

Inverse kinematics computes the angles of the servo motors that are required to perform the desired movement. These angles are in radians and should be transformed to the corresponding number within the I2c range (from 1 to 255).

Thigh servo responsible for the rotation in a horizontal plane:

The computed angles range from $-\pi/2$ to $\pi/2$. Angle value of $-\pi/2$ radians correspond to number 1 written to the register, angle value of $\pi/2$ radians corresponds to number 255 written to the register.

$$\frac{I2crange}{angle,range} = \frac{254}{\pi} \tag{24}$$

In order for the previous to be fulfilled, we attach the following block set between the I2c writing block and the result of inverse kinematics.

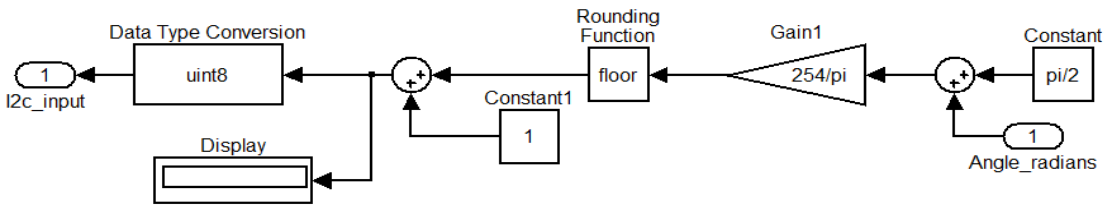


Figure (23) Connecting inverse kinematics to the I2c writing block

The previous set of blocks is also used for the knee servo, but with a different I2c range (212) that corresponds to the allowed angle rotation, which is about $5\pi/6$ due to the mechanical limitations, which make us sure that the motor will not hit its stop.

Thigh servo responsible for the rotation in a vertical plane:

When using the previous set of blocks, the motion of the mentioned servo was reversed. This indicated that angle value of $-\pi/2$ radians corresponds to number 255 written to the register, and angle value of $\pi/2$ radians corresponds to number 1 written to the register. To achieve this, the previous set of blocks was modified in the following way:

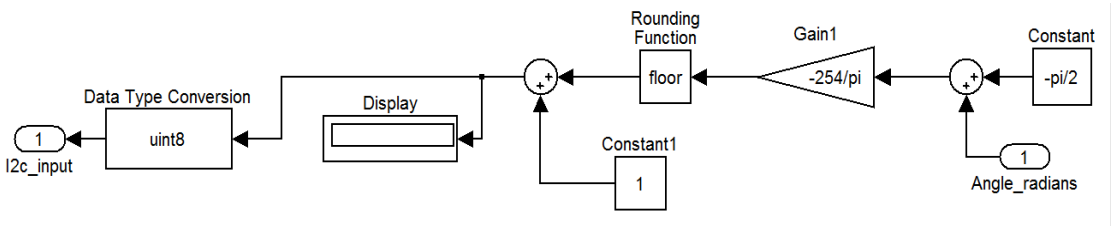


Figure (24) Modified connection of inverse kinematics to I2c writing block

3.5.2 Using interpolation to program some basic movements:

In order to accomplish any desired movement by the robot, these steps can be followed: 1- Choosing the rotation angle of the servo motor to accomplish the desired movement. 2- Calculating the corresponding X,Y,Z coordinates of the end effecter. 3- The previous X,Y,Z coordinates are fed with the corresponding time vector to the repeated sequence interpolating blocks, whose outputs are fed to the inverse kinematics model.

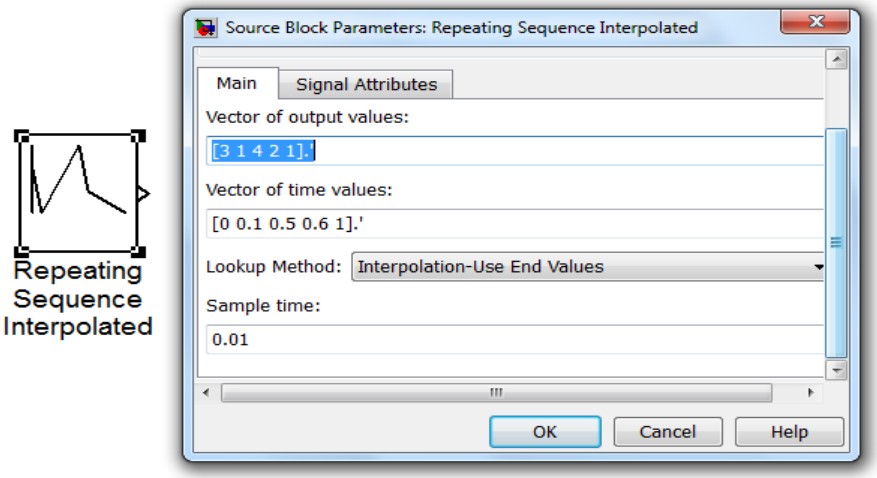


Figure (25) Repeating sequence interpolated block and its parameters

Note1: The previous block uses linear interpolation between the values of the output vector, which is not suitable when it comes to initial and final velocities or accelerations. In order to minimize them we can use the following two vectors:

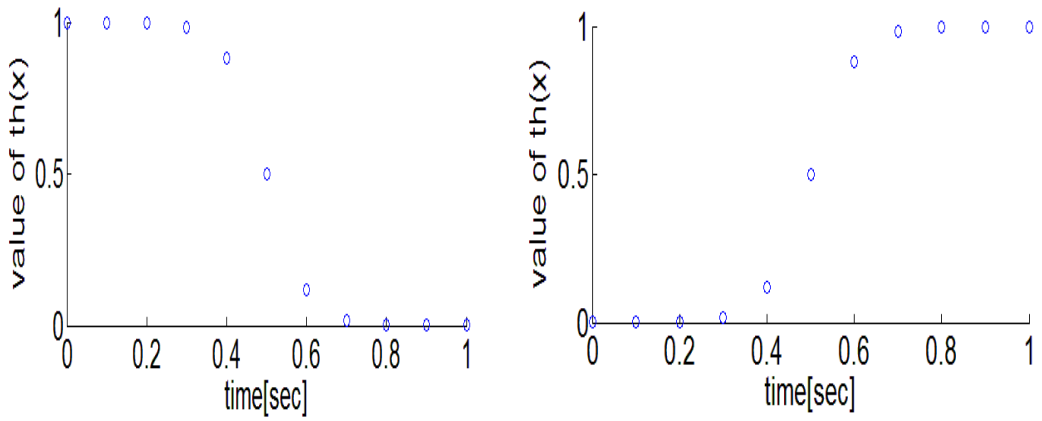


Figure (26) Data vectors used to minimize initial and final velocities and acceleration.

These vectors are generated by using the hyperbolic tangent function in Matlab, the combination of them multiplied with the right coefficients is used to form the vector of the output values. Data vector to the left is called Y, while data vector to the right is called Y1.

Note2: Although we know the desired rotation angles, inverse kinematics is used in order to get a smooth movement. We want the position to be a combination of Y and Y1 vectors, and the use of inverse kinematics makes sure that the rotation angles correspond to it.

Note3: The initial coordinates of each leg are fed to the kinematic model, so the output values of the sequence interpolated block represent the difference between the desired value and the initial coordinate.

Note4: VRML model is attached to inverse kinematics which enables to verify the results without the use of the robot.

3.5.3 Standing up, turning around and sitting algorithms:

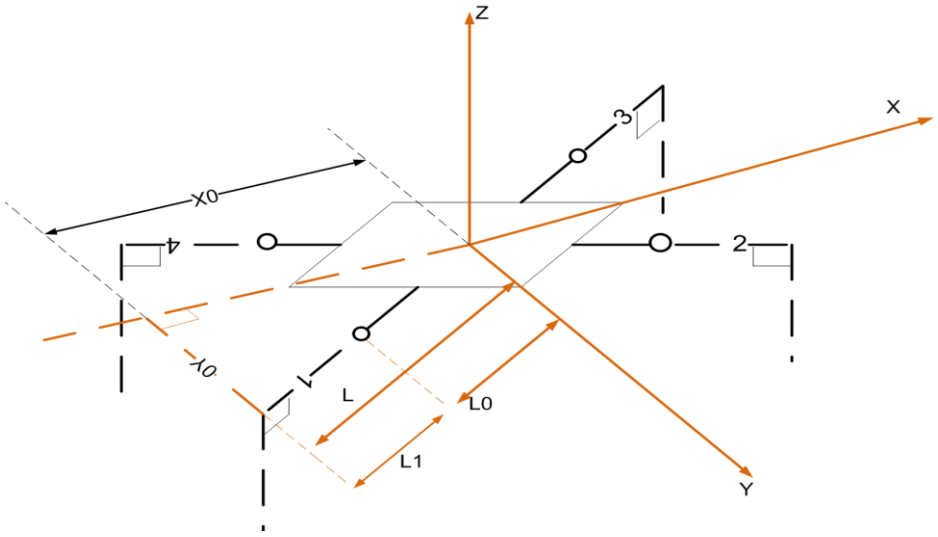


Figure (27) Rough sketch of the robot

In order for the robot to stand up all legs are to be lifted, and then lowered with certain knee angle, so the robot would push itself up and its body would move upwards. In the lifting mode, the change of Z coordinates should be positive, and after a number of tries it is set to about 0.044m. In the sitting down mode, actions are reversed.

The turning around algorithm is a bit more complicated, and it involves some calculations. We choose a rotation angle of 45 degrees of each leg, then we calculate the changes in the X, Y coordinates of each leg. First L, L1, L0 should be calculated or measured:

$$L = \sqrt{X_0^2 + Y_0^2} = \sqrt{2.(0{,}1357)^2} = 0.191m \quad (25)$$

$$L_0 = 0.07m \quad \text{(By measuring)} \quad (26)$$

$$L_1 = L - L_0 = 0.121m \quad (27)$$

L : is the length of the robot leg from the coordinate system origin to the knee when the thigh is in a horizontal position.

L_0 : The Length of the fixed part of L .

L_1 : The length of the movable part of L .

X_0, Y_0 : Initial X,Y coordinates of the end effecter that are included in the starting M-file.

3.5.3.1 Calculation of the change in the X, Y coordinates of each leg:

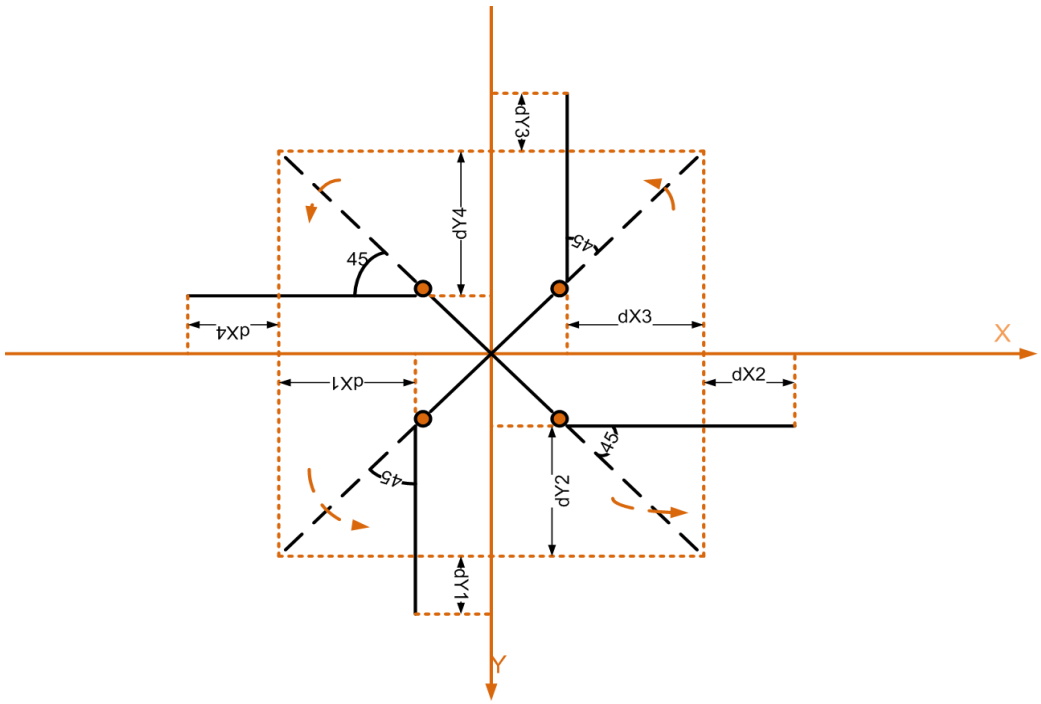


Figure (28) Representation of changes of X, Y coordinates while rotating

Leg number 1:

$$dx_1 = L.\sin 45 - L_0.\sin 45 = 0.135 - 0.049 = 0.086m \quad (28)$$

$$dy_1 = L_1 + L_0.\cos 45 - L\cos 45 = -0.035m \quad (29)$$

Leg number 2:

$$dx_2 = L_1 + L_0.\cos 45 - L.\cos 45 = 0.035m \quad (30)$$

$$dy_2 = L_0.\sin 45 - L.\sin 45 = -0.086m \quad (31)$$

Leg number 3:

$$dx_3 = L_0.\cos 45 - L.\cos 45 = -0.086m \quad (32)$$

$$dy_3 = -L_1 - L_0.\cos 45 + L.\sin 45 = -0.035m \quad (33)$$

Leg number 4:

$$dx_4 = -L_0.\sin 45 - L_1 + L - \sin 45 = -0.035m \quad (34)$$

$$dy_4 = -L_0.\cos 45 + L.\cos 45 = 0.086m \quad (35)$$

dx : change along the X axis.

dy : change along the Y axis.

Note1: The previous calculated changes of the Y coordinate should be countered, so we get the desired movement. The sign of the previous Y changes should be inversed.

Note2: As mentioned before, data vectors Y, Y1 are used to enter the changes to the output field in the repeated sequence interpolated block.

$\frac{Y}{11.63} = 0 \to 0.086; \frac{Y}{28.57} = 0 \to 0.035$.Y1 divided by the same values is used to counter these changes.

$\frac{Y1-1}{11.63} = 0 \to -0.086; \frac{Y1-1}{28.57} = 0 \to -0.035$. Y-1 divided by the same values is used to counter these changes.

Note 3: Before rotation, each leg is lifted off the ground by changing the corresponding Z coordinate (dZ=0.032). After rotation is done, the leg is lowered to touch the ground. To lift the Leg, we use the vector **Y** divided by 31.25. For lowering, we use the vector **Y1** also divided by 31.25.

Note4: After rotating all legs each at a time, we rotate all legs at once in the opposite direction while the legs keep contact with the ground, which makes the body of the robot rotate. The same calculated changes of the coordinates are used, but with an opposite sign.

3.5.4 Walking algorithm:

This algorithm that will be explained in the following few pages has been inspired by a successful project called X-Walker [15]. One of the problems that this project has dealt with is designing a walking algorithm for a four legged robot.

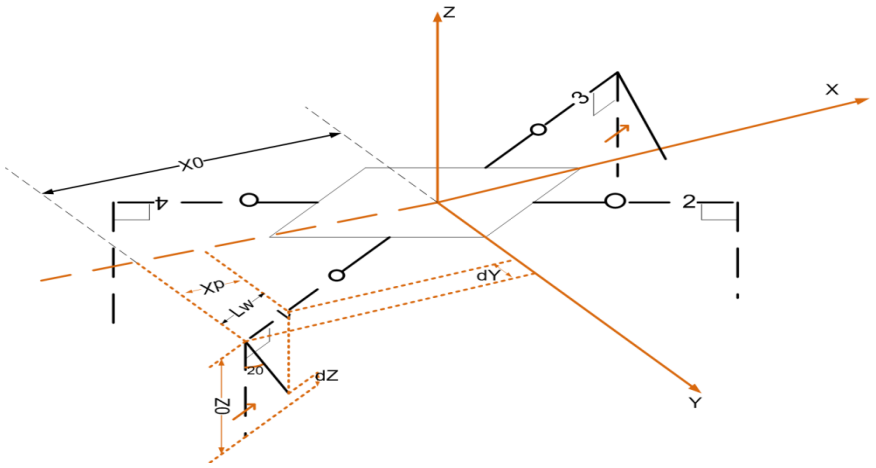


Figure (29) Changes of X, Y, Z coordinates while walking

$$L_w = |Z_0|.\cos 70 = 0.046m \quad (36)$$

$$dx = -(L - L_W).\cos 45 + L.\cos 45 = 0.033m \quad (37)$$

$$dy = (L - L_w).\sin 45 - L.\sin 45 = -0.033m \quad (38)$$

$$dz = -|Z_0|.\sin 70 + |Z_0| = 0.008m \quad (39)$$

dx, dy, dz : Changes along the X,Y,Z axes.

The calculated changes in the Y,Z coordinates should be countered in order to keep the leg in touch with the ground. These results besides the ones reached in the turning around algorithm enable us to calculate changes of X, Y, Z coordinates that will lead to the walking of the robot. Through the following four steps that illustrate the walking algorithm, X,Y,Z changes will not be calculated, but we will use analogy to figure out the needed values.

Note: The following algorithm aims to move the robot along the Y axis.

3.5.4.1 Step number one of the walking algorithm:

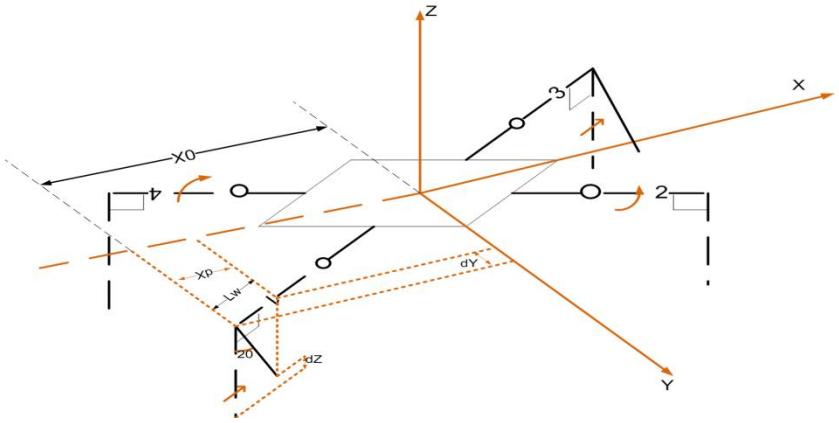


Figure (30) Step one of the walking algorithm

During this step the body of the robot will move towards leg number 1, and this can be done by bending the legs 1 and 3, and rotating the legs 2 and 4 in the directions showed in the figure above. During this step all legs keep in touch with the ground. Here are the X,Y,Z changes of each leg:

Leg number	dx[m]	dy[m]	dz[m]
1	0.033	0.033	-0.008
2	0.035	0.086	0
3	0.033	0.033	-0.008
4	0.086	0.035	0

Note1: To apply the changes of dx, dy of legs 1 and 3, we use vector Y /30.3, whereas for dz of the same legs, vector Y/125 is used.

Note2: Legs 2 and 4 are rotated by 45 degrees.

3.5.4.2 Step number two of the walking algorithm:

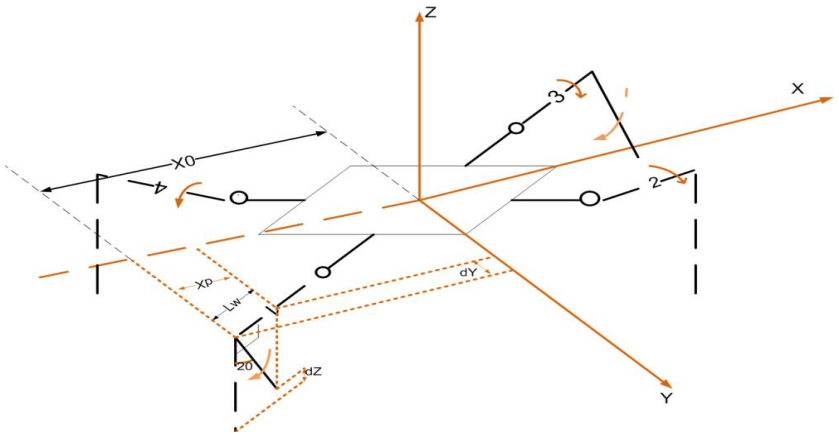


Figure (31) Step number two of the walking algorithm

This step consists of three sub stages. At the first sub stage, leg number three is lifted, rotated by 45 degrees then lowered. While lifting the leg, it is repositioned into the original position to form a right angle. At the second sub stage, Leg number 2 is lifted, rotated by 45 degrees then lowered, at the same time, leg number 1 is repositioned to the original right angle. At the third stage, leg four is lifted, rotated then lowered. Here are the changes of X,Y,Z coordinates of each leg:

Leg number	Time	dx[m]	dy[m]	dz[m]
3	t1	0	0	0.032
3	t2	0.035	0.086	0.032
3	t3	0.035	0.086	0
2	t4	0.035	0.086	0.032
2	t5	0	0	0.032
2	t6	0	0	0
1	t4	0	0	0
4	t7	0.086	0.035	0.032
4	t8	0	0	0.032
4	t9	0	0	0

Note1: Every value of dx,dy,dz at a given time represents the change of the given coordinate from the original value, which is implemented in the starting M-file. This means that when this change remains constant through different intervals, no move is desired, but when it differs, a certain move that corresponds to the change will take place, which implies that 0 values that are in red color correspond to certain movement of the specified leg.

3.5.4.3 Step number three of the walking algorithm:

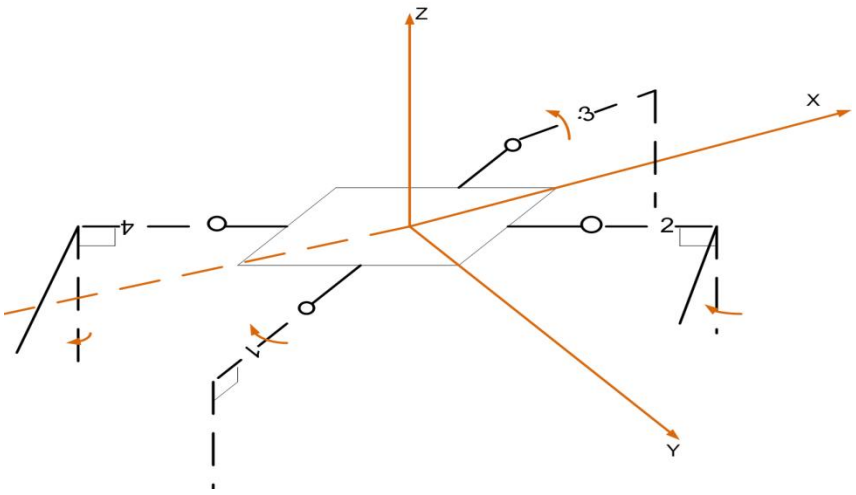


Figure (32) Step number three of the walking algorithm

During this step, legs three and one are rotated by 45 degrees while touching the ground, and at the same time legs two and four are bent ,which makes the body of the robot move in the direction of leg number two. Here are the corresponding changes of the X,Y,Z coordinates:

Leg number	dx[m]	dy[m]	dz[m]
1	-0.035	0.086	0
2	-0.033	0.033	-0.008
3	-0.035	0.086	0
4	-0.033	0.033	-0.008

3.5.4.4 Step number four of the walking algorithm:

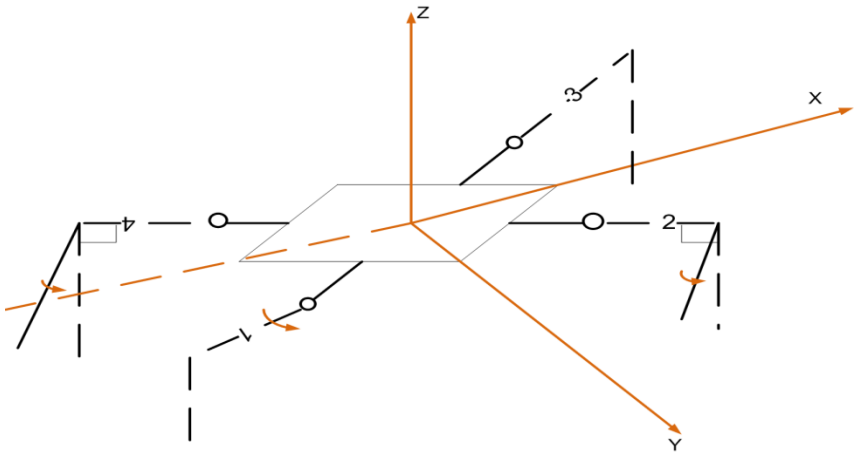


Figure (33) Step number four of the walking algorithm

The goal of this step is to return the robot to its original position, so another walking cycle could be initialized. Leg 1 is lifted, rotated by 45 degrees then lowered, and at the same time legs 2 and 4 are returned to their original positions. Here are the changes of X,Y,Z coordinates of each leg:

Leg number	dx[m]	dy[m]	dz[m]
1	0	0	0
2	0	0	0
4	0	0	0

4 Conclusions:

After considering a number of sensors that would enable the robot to receive data about its status, current sensing resistors have been chosen and implemented into the control unit. These sensors measure current consumed by each servo motor, and feed the microcontroller with this data to form estimation about the situation the robot is facing. In addition to the current sensing resistors, the servo driver and the microcontroller form the main parts of the designed control unit. The microcontroller receives data from the sensors, and sends instructions to the servo driver to drive the motors of the robot according to the downloaded program. The previous unit has been tested through implementing a number of algorithms into the microcontroller. These algorithms enable the robot to do some simple movements such as standing up, sitting down, turning around and walking, which also confirms that the control unit is functional and can serve as a good base for any upcoming job that will use the acquired data by the sensors to build more complex algorithms.

5 References:

[1] GREPL, Robert. *Kinematika a dynamika mechatronických systémů*. Vyd. 1. Brno: Akademické nakladatelství CERM, 2007, 158 s. ISBN 978-80-214-3530-8.

[2] HEROUT, Pavel. *Učebnice jazyka C*. 4., přeprac. vyd. České Budějovice: Kopp, 2004, 271, viii s. ISBN 80-723-2220-6.

[3] GREPL, Robert. *Modelování mechatronických systémů v Matlab SimMechanics*. 1. vyd. Praha: BEN - technická literatura, 2007, 151 s. ISBN 978-80-7300-226-8.

[4] VOREL, Pavel a Patočka,Miroslav . Průmyslová elektronika [online]. [cit. 2013-05-13]. Available at:: https://www.vutbr.cz/www_base/priloha.php?dpid=217

[5] VOREL, Pavel a Procházka,Petr. *Řídící členy v elektrických pohonech* [online]. [cit. 2013-05-13]. Available at:: https://www.vutbr.cz/www_base/priloha.php?dpid=21765

[6] NATIONAL INSTRUMNETS. *Strain gauge measurments - A tutorial* [online]. 1998 [cit. 2013-05-13]. Available at:: http://www.eidactics.com/Downloads/Refs- Methods/NI_Strain_Gauge_tutorial.pdf

[7] HONEYWELL. *Force Sensors FSS Low Profile Force Sensors* [online]. 2003 [cit. 2013-05-13]. Available at:: http://www.farnell.com/datasheets/30736.pdf

[8] SPECTRA SYMBOL. *Flex Sensor* [online]. [cit. 2013-05-13]. Available at: https://www.sparkfun.com/datasheets/Sensors/Flex/flex22.pdf

[9] KISTLER. Force – FMS [online]. [cit. 2013-05-13]. Available at: http://www.intertechnology.com/Kistler/pdfs/Force_Model_9251A_9252A_9250A4_9251A4.pdf

[10] ALLEGRO. ACS 712 [online]. [cit. 2013-05-13]. Available at:: http://www.allegromicro.com/~/media/Files/Datasheets/ACS712-Datasheet.ashx

[11] ANALOG DEVICES. *AD8170/AD8174* [online]. [cit. 2013-05-13]. Available at:: http://www.analog.com/static/imported-files/data_sheets/AD8170_8174.pdf

[12] VISHAY. DG4051A/DG4052A/DG4053A [online]. [cit. 2013-05-13]. Dostupné z: http://pdf1.alldatasheet.com/datasheet-pdf/view/250122/VISHAY/DG4052A.html

[13] I2C. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-05-13]. Available at: http://en.wikipedia.org/wiki/I²C

[14] SD20 - 20 Channel I2C to Servo Driver Chip The SD20 [online]. [cit. 2013-05-13]. ISBN 98-80-214-3530-8. Available at:

http://blog.mlrobotic.com/wp-content/uploads/2009/05/SD20.pdf

[15] X-walker quadruped robot [online]. 2011 [cit. 2013-05-13]. Available at: http://www.youtube.com/watch?v=3Q0Lgf-Zaug

6 Appendix:

[App.1] Boards and schematics of the control unit

[App.2] Starting M-file and Simulink models illustrating the designed algorithms in Matalb 2012a version using VRML.

[App.3] Starting M-file and Simulink models used for implementing the designed algorithms into the microcontroller.

[App.4] Youssef_Daniel_bachelor_thesis_EN.pdf